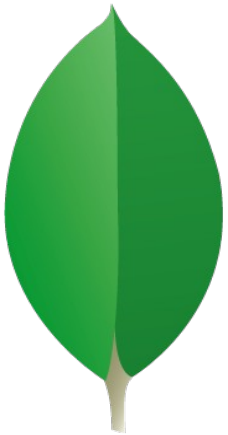
pro-netics group

sourcesense
making sense of open source



mongoDB

Dessì Massimiliano

SpringFramework Meeting 12 settembre 2009 Cagliari

Author

Software Architect and Engineer
ProNetics / Sourcesense

Presidente
JugSardegna Onlus

Fondatore e coordinatore
SpringFramework Italian User Group

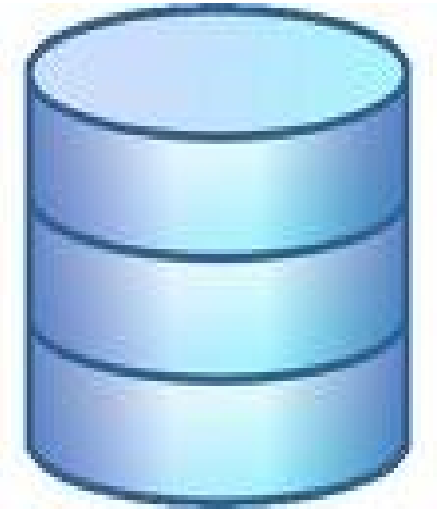
Committer
OpenNMS - MagicBox

Autore
Spring 2.5 Aspect Oriented Programming

pro-netics group

sourcesense
making sense of open source

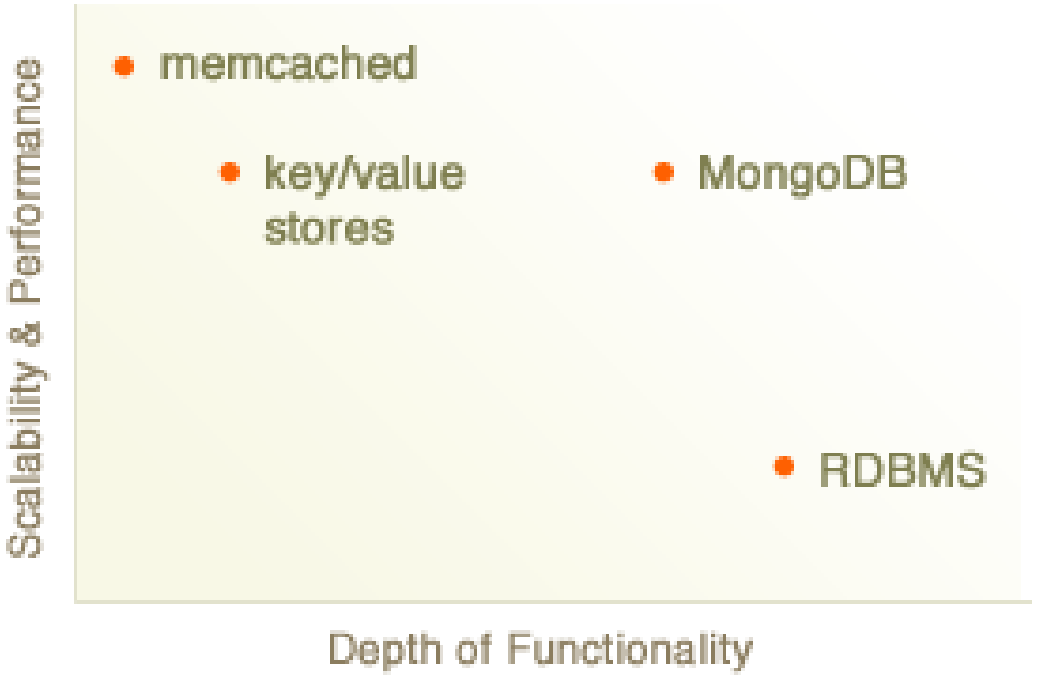
Abbiamo ancora bisogno di un RDBMS ?



pro-netics group

sourcesense
making sense of open source

Dipende



pro-netics group

sourcesense
making sense of open source

Neo4j

Sesame

FreeBase

Graph databases

Hypertable

DirectedEdge

BigTable

AllegroGraph

Cassandra

HBase

MongoDB

CouchDB

Document databases

ThruDB

JackRabbit

Voldemort

Scalaris

Dynomite

TokioCabinet

Distributed key-value stores

Skynet

Hadoop

Ringo

MapReduce

MemcacheDB

Disco

pro-netics group

sourcesense
making sense of open source

BigTable, Google 2006

http://labs.google.com/papers/bigtable.html

Le colonne non sono predefinite

Le righe possono esser aggiunte con qualsiasi numero di colonne

Le colonne vuote non vengono salvate

Adatto alle applicazioni dove gli attributi degli oggetti non sono conosciute o cambiano frequentemente

pro-netics group

sourcesense
making sense of open source

Graph database

Le relazioni tra gli item sono la caratteristica principale

Modello dei dati molto interconnesso

Adatto se nelle applicazioni avete delle lunghe catene di join, molte relazioni many to many e i dati sono già sotto una forma di grafo (social net..).
I graph database sono spesso associati al web semantico e ai datastore RDF.

pro-netics group

sourcesense
making sense of open source

MapReduce, Google 2004

http://labs.google.com/papers/mapreduce.html

Lavori batch di grandi moli di dati senza preoccuparsi della infrastruttura.

- Automatic parallelization and distribution
- Fault-tolerance
- I/O scheduling
- Status and monitoring

Adatto alle applicazioni dove si processando molti dati in processi batch.

pro-netics group

sourcesense
making sense of open source

Distributed key-value store

I database distribuiti partizionano e replicano i dati in maniera trasparente i dati su molte macchine in un cluster.
I dati possono non essere immediatamente consistenti.

La scelta è tra:

- Bassa latenza (velocità per req-res)
- Alto throughput (processi batch)

Adatti ad applicazioni dove i dati sono indipendenti e la disponibilità e performance dei dati sono più importanti delle caratteristiche ACID.

pro-netics group

sourcesense
making sense of open source

Document database

Gli item sono dei Documenti
Non sono permesse joins e transactions spalmate su più righe o documenti.
Un documento è formato da valori o liste in formato JSON o XML e viene lavorato sulla struttura di un documento.
Estrazione, indicizzazione aggregazione e filtraggio sono basati sugli attributi contenuti nel documento

Adatto alle applicazioni dove i dati sono indipendenti tra loro e non sono richieste join.

pro-netics group

sourcesense
making sense of open source

MongoDB

http://www.mongodb.org

DocumentDB

Veloce

Schema Free

Query dinamiche

Query Profiling

Dati sono salvati in formato binario BSON

Dati organizzati in Documenti o Collezioni in JSON

Efficiente salvataggio di contenuti binari anche grandi

pro-netics group

sourcesense
making sense of open source

MongoDB

Full index support

Replication and fail-over support

Auto-sharding for cloud-level scalability

Caratteristiche di un key-value e di un RDBMS

Alta scalabilità

Supporto commerciale

p.s. Possiamo fare il dump :-)

pro-netics group

sourcesense
making sense of open source

Adatto

Siti web

High volume, low data

Alta scalabilita

Dati in formato JSON

Caching

Logging

Analisi real-time

pro-netics group

sourcesense
making sense of open source

Non Adatto

Sistemi altamente transazionali

Traditional Business Intelligence

Problemi che richiedono il SQL

pro-netics group

sourcesense
making sense of open source

Documento != Stringa

In prima istanza si potrebbe pensare di avere un Document DB salvando stringhe dentro le righe di un RDBMS.

Un Document DB nasce invece sul concetto di documento e collezione, non di righe e colonne, infatti il modo di salvare e di recuperare le informazioni è diverso

pro-netics group

sourcesense
making sense of open source

Document struttura JSON


```
{ "_id" : "027b6e279578a64aa0684700" , "address-city" : "Ca" ,
"address-zipCode" : "09100" , "address-province" : "CA" ,
"address-region" : "Sardegna" , "address-country" : "Campidano" ,
"address-streetName" : "V.le Europe" , "address-streetNumber" : "4" ,
"telephone-contactMobile" : "3391234556" ,
"telephone-contactTelephoneHome" : "070123456" ,
"telephone-contactTelephoneWork" : "070987654" ,
"telephone-contactAcceptSms" : true ,
"userInfo-dateOfBirth" : "2009-09-08T15:30:30Z" ,
"userInfo-email" : "max@gmail.com" , "userInfo-name" : "Paperino" ,
"userInfo-surname" : "Paolino" , "userInfo-sensibleData" : "no sensible data" ,
"id" : "d37m3051128" , "description" : "descr" , "groupId" : "15" ,
"centerId" : "centerThree" , "_ns" : "centerUser"}
```

pro-netics group

sourcesense
making sense of open source

Schema Free

In un RDBMS la struttura dei dati è vincolata allo schema in cui definiamo tabelle con le colonne per meglio relazionare i dati.

In MongoDB lo “schema” viene definito dai dati quando vengono salvati.

Raggruppiamo le entità della nostra applicazione in collezioni (Users, Centers...).

Document

Le informazioni possono essere embedded o referenziate.

Ovviamente la soluzione embedded è più veloce non avendo bisogno di caricare altri dati come farebbe una join con il SQL.

Automaticamente ogni Document ha un campo _id per identificarlo univocamente.

pro-netics group

sourcesense
making sense of open source

Atomicità

Il motivo principale della mancanza delle transazioni è la lentezza e il costo dei lock in ambienti distribuiti.

Non essendoci le transazioni che coinvolgono più dati, qui abbiamo atomicità a livello di singolo documento con i seguenti comandi:

```
$set  $inc $push $pushAll $pull $pullAll
```

Mongo Driver

Mongo può essere utilizzato con i linguaggi più diffusi:

Java (Groovy, Scala, JRuby), PHP, Ruby, C++, Python, Perl, C#, Erlang, Javascript server side.

Ogni driver predispone i nostri oggetti ad essere utilizzati in Mongo.

Collection

I nostri item (Document) sono salvati dentro delle Collection che possiamo vedere come i corrispettivi delle tabelle in un RDBMS.

Una collection viene creata effettivamente quando un document viene salvato al suo interno.

```
public void createCollectionCenters(Mongo mongo) {
     NoCenter center = new NoCenter();
     DBCollection collectionCenters = mongo.getCollection("centers");
     collectionCenters.insert(new BasicDBObject(center.toMap()));
 }
```

pro-netics group

sourcesense
making sense of open source

Mongo Java Driver

Per essere salvati o letti i nostri oggetti devono essere “traslati” in oggetti `com.mongodb.DBObject`, per fare questo ci sono due alternative:

Implementare l' interfaccia `DBObject`

Utilizzare un `BasicDBObjectBuilder`

Mongo Java Driver

Se la nostra `MyClass` implementa l' interfaccia `DBObject`

assegneremo i valori in questo modo.

```
MyClass myObj= new MyClass();
myObj.put("user", userId);
myObj.put("message", msg);
myObj.put("date", new Date());
...
collection.insert(myObj);
```

pro-netics group

sourcesense
making sense of open source

Mongo Java Driver

Utilizzando BasicDBObjectBuilder

```
collection.insert(
   BasicDBObjectBuilder.start()
   .add("user", myObj.getId())
   .add("user", myObj.getUser())
   .add("date", new Date())
   .....
   .get();
);
```

Mongo Java Driver

Una terza alternativa, più pratica consiste nel passare al Builder la mappa con i valori contenuti nel nostro oggetto.

```
BasicDBObjectBuilder.start(myObj.toMap()).get();
```

Dobbiamo semplicemente aggiungere agli oggetti della nostra applicazione una rappresentazione sotto forma di Map e un costruttore con un Map per ricostruirli da un `DBObject`

```
public class MyClass{

    public MyClass(Map map){...}

    public Map toMap(){..}
}
```

Concorrenza Web App

Nel caso di WebApp dove si hanno pesanti carichi in scrittura, è oppurtuno per avere consistenza nella sessione di lavoro (HttpRequest), eseguire il lavoro come mostrato nel codice, magari usando l' AOP con un before advice dove chiamare lo start e un after advice dove chiamare il done.

```
Mongo dataStore;
dataStore.requestStart();

//your code

dataStore.requestDone();
```

Interrogazioni

Le query di ricerca vengono fatte costruendo i parametri di ricerca tramite un `DBObject`.

Possiamo anche paginare il risultato della query.

```
List<DBObject> objects =
   collection.find(
      BasicDBObjectBuilder.start().
         add("userInfo-surname", surname).
         add("centerId", centerId).get()
   ).
   skip(offset * page).limit(offset).toArray();
```

pro-netics group

sourcesense
making sense of open source

Interrogazioni

Numero documenti in una collezione

```
collection.find(
   BasicDBObjectBuilder.start()
   .add("centerId",centerId).get()
).length();
```

Interrogazioni

Se vogliamo recuperare solo alcuni campi del documento (Select SQL), costruiamo un `DBObject` con quei campi e utilizziamo il find che accetta due DBObject (query, campi richiesti).

Il `DBCursor` è il corrispondente del ResultSet jdbc.

```
DBCursor cursor = coll.find(
    BasicDBObjectBuilder.start().add("centerId", centerId).get(),
    BasicDBObjectBuilder.start().add("userInfo-email", "").get()
);
```

Interrogazioni

Con espressioni regolari

```
String pattern = new StringBuilder().append(character).append("*").toString();
List<DBObject> objects = coll.find(BasicDBObjectBuilder.start()
                    .add("userInfo-surname",
                          java.util.regex.Pattern.compile(pattern))
                    .get()).toArray();
```

pro-netics group

sourcesense
making sense of open source

Interrogazioni avanzate

Nelle query abbiamo a disposizione anche :

`<, <=, >, >=, $ne, $in, $nin, $mod, $all, $size, $exists`

Valori dentro Array

Valori dentro un oggetto embedded

Full language expressions con $where

`limit(), skip(), snapshot(), count(), group()`

pro-netics group

sourcesense
making sense of open source

Insert

```
public String insertCenterUser(CenterUser user) {
    Map fields = user.toMap();
    fields.put("id", IdGenerator.getUniqueId());//CustomId
    DBObject obj = coll.insert(BasicDBObjectBuilder.start(fields).get());
    return obj != null ? obj.get("id").toString() : Constants.ID_NEW;
}
```

pro-netics group

sourcesense
making sense of open source

Update

```
public int updateCenterUser(CenterUser user) {

        DBObject  obj = coll.update(
                    BasicDBObjectBuilder.start().
                       add("id",user.getEntity().getId()).get(),
                    BasicDBObjectBuilder.start(user.toMap()).get(),
                    false,
                    false
                );
         return obj != null ? 1 : 0;
}
```

pro-netics group

sourcesense
making sense of open source

Upsert

Update if present insert is missing

```
public int updateCenterUser(CenterUser user) {

      DBObject obj = coll.update(
                  BasicDBObjectBuilder.start().
                     add("id",user.getEntity().getId()).get(),
                  BasicDBObjectBuilder.start(user.toMap()).get(),
                  true,
                  false
               );
       return obj != null ? 1 : 0;
}
```

pro-netics group

sourcesense
making sense of open source

Delete

Delete diretto

```
public void deleteCenterUser(String id, String centerId) {
    DBObject obj = coll.findOne(
          BasicDBObjectBuilder.start().
          add("id", id).
          add("centerId", centerId).get());
    coll.remove(obj);
}
```

Delete

Delete utilizzando il DBcursor

```
public boolean deleteCenterUser(String id, String centerId) {
        DBCursor cursor = coll.find(
                          new BasicDBObject(),
                           BasicDBObjectBuilder.start()
                           .add("id", id)
                           .add("centerId", centerId).get()
                          );
        boolean result = false;
        while (cursor.hasNext()) {
            coll.remove(cursor.next());
            result = true;
        }
        return result;
}
```

pro-netics group

sourcesense
making sense of open source

Production Deployments

http://www.mongodb.org/display/DOCS/Production+Deployments

SourceForge Mozilla Ubiquity Disqus

Business Insider Floxee

Silentale BoxedIce Defensio

TweetSaver ShopWiki MusicNation Detexify

Sluggy Freelance eFlyover soliMAP

(Sept 2009)

pro-netics group

sourcesense
making sense of open source

Domande ?

pro-netics group

sourcesense
making sense of open source

Grazie per l'attenzione !

Massimiliano Dessì
desmax74 at yahoo.it
massimiliano.dessi at pronetics.it
http://twitter.com/desmax74
http://jroller.com/desmax
http://www.linkedin.com/in/desmax74
http://wiki.java.net/bin/view/People/MassimilianoDessi
http://www.jugsardegna.org/vqwiki/jsp/Wiki?MassimilianoDessi

Spring Framework Italian User Group
http://it.groups.yahoo.com/group/SpringFramework-it